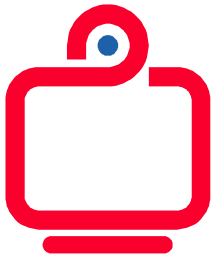


Instruction-level Parallelism in AES Candidates

- 2nd AES Candidate Conference, Rome, March 1999

Craig Clapp

PictureTel Corporation

Evaluation methods

- Theoretical performance limits
  - Critical-path analysis
- Practical performance
  - C-code performance on a family of RISC / VLIW CPUs having 1 to 8 execution units

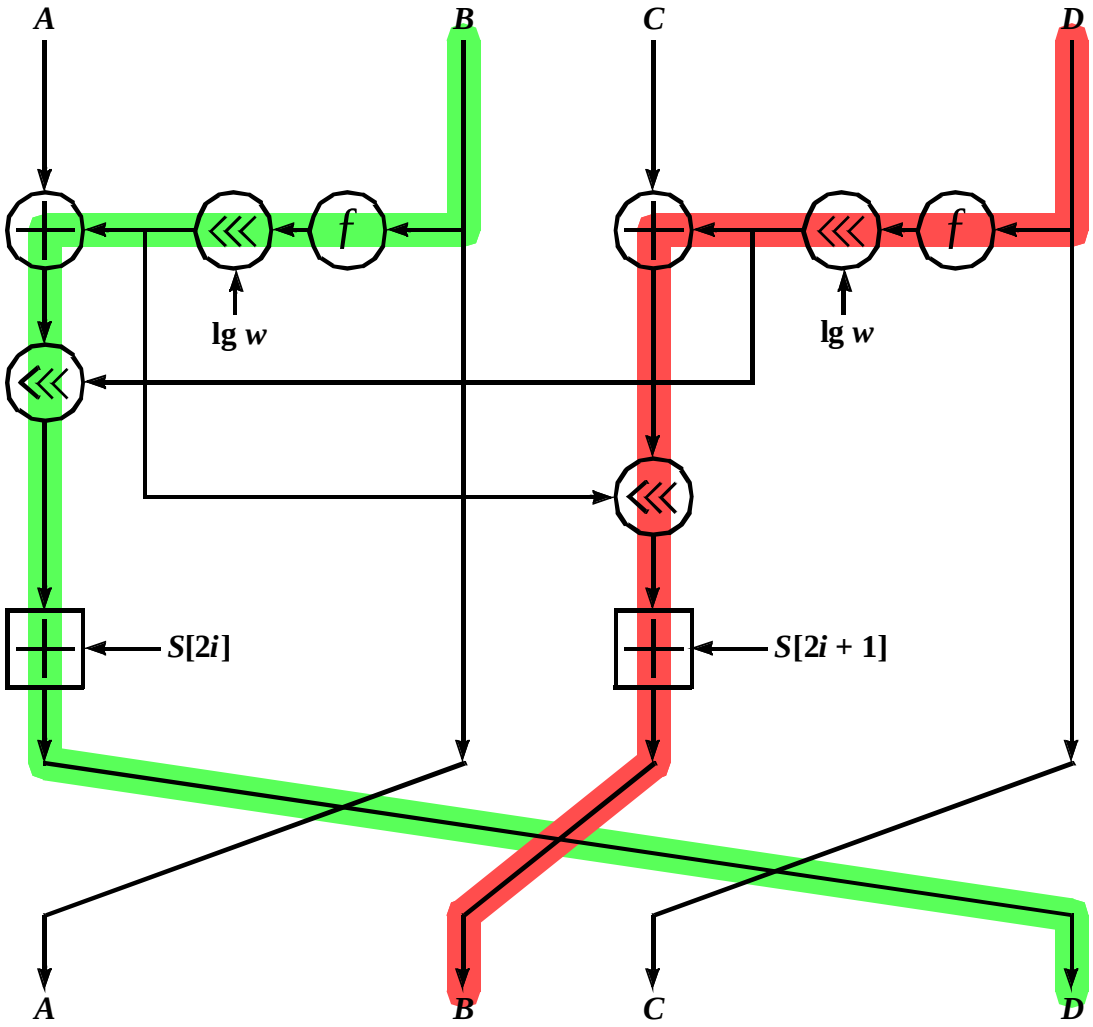
Candidates studied

- Crypton
- E2
- Mars
- RC6
- Rijndael
- Serpent
- Twofish

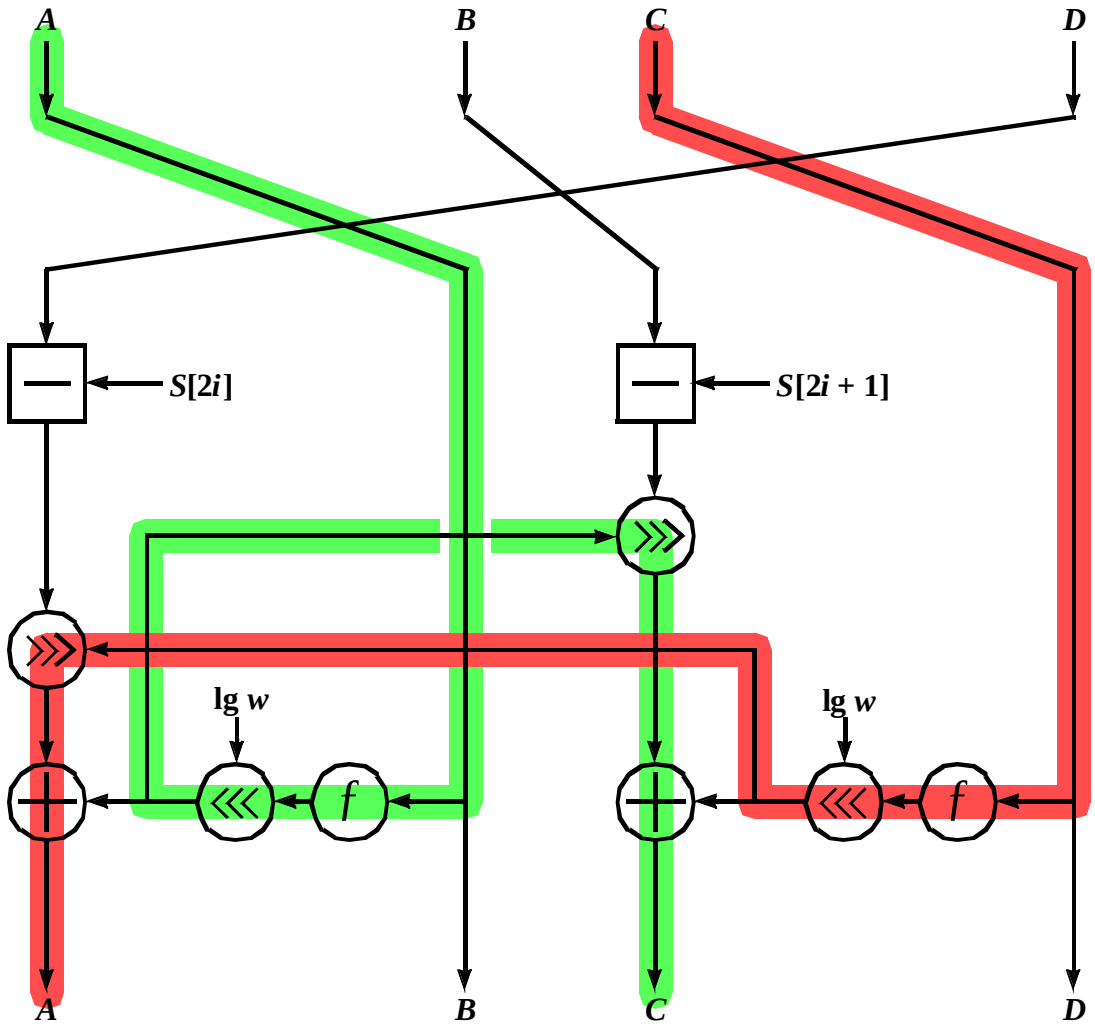
RISC / VLIW CPU instruction set

- Single-cycle instructions
  - Add, Sub, AND, OR, XOR, NOT
  - Shift, Rotate, Byte Extract, Store
- Multi-cycle instructions (pipelined)
  - Load (3 cycles)
  - 32 x 32 multiply (3 cycles)

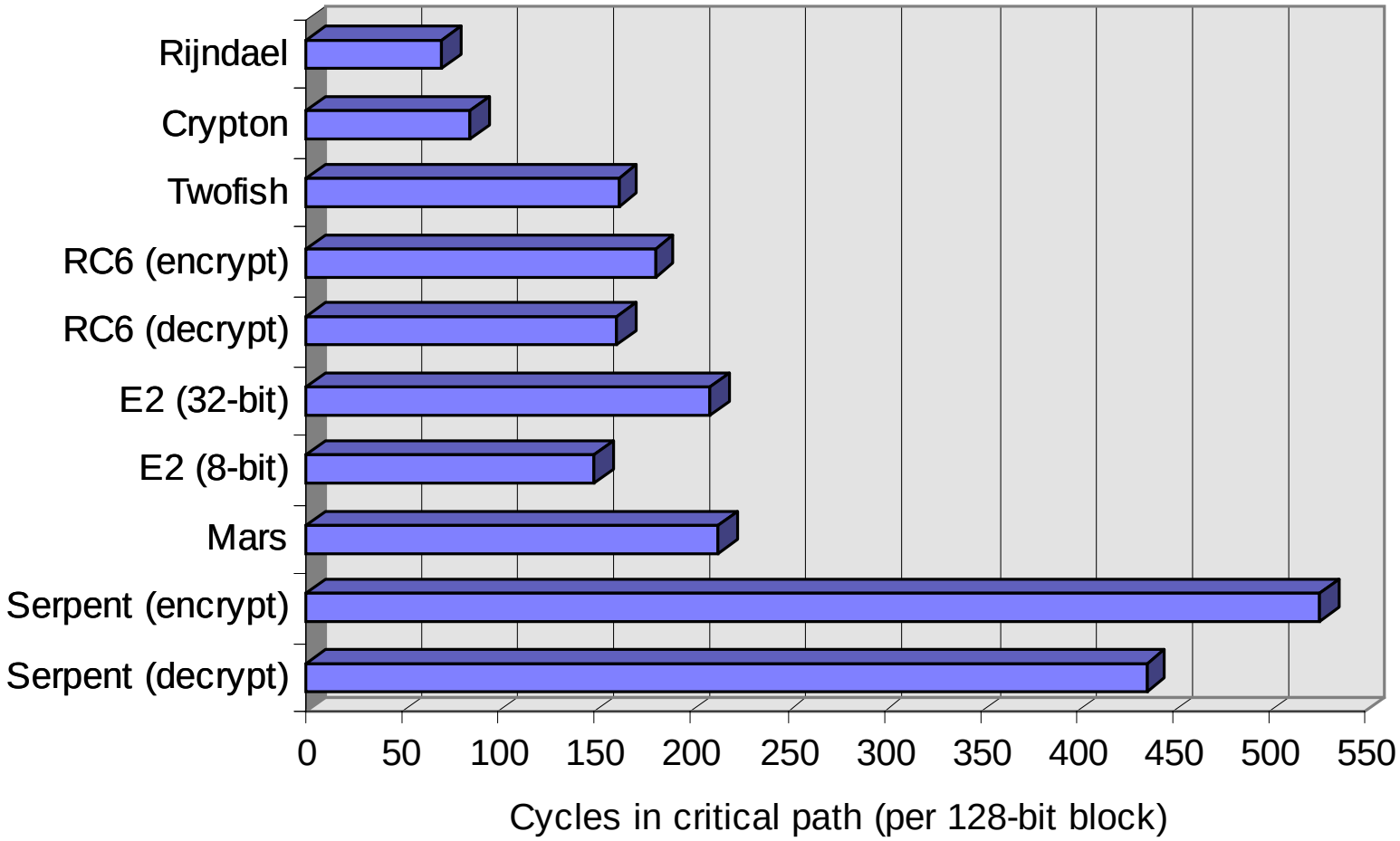
Critical-path example: RC6 encryption



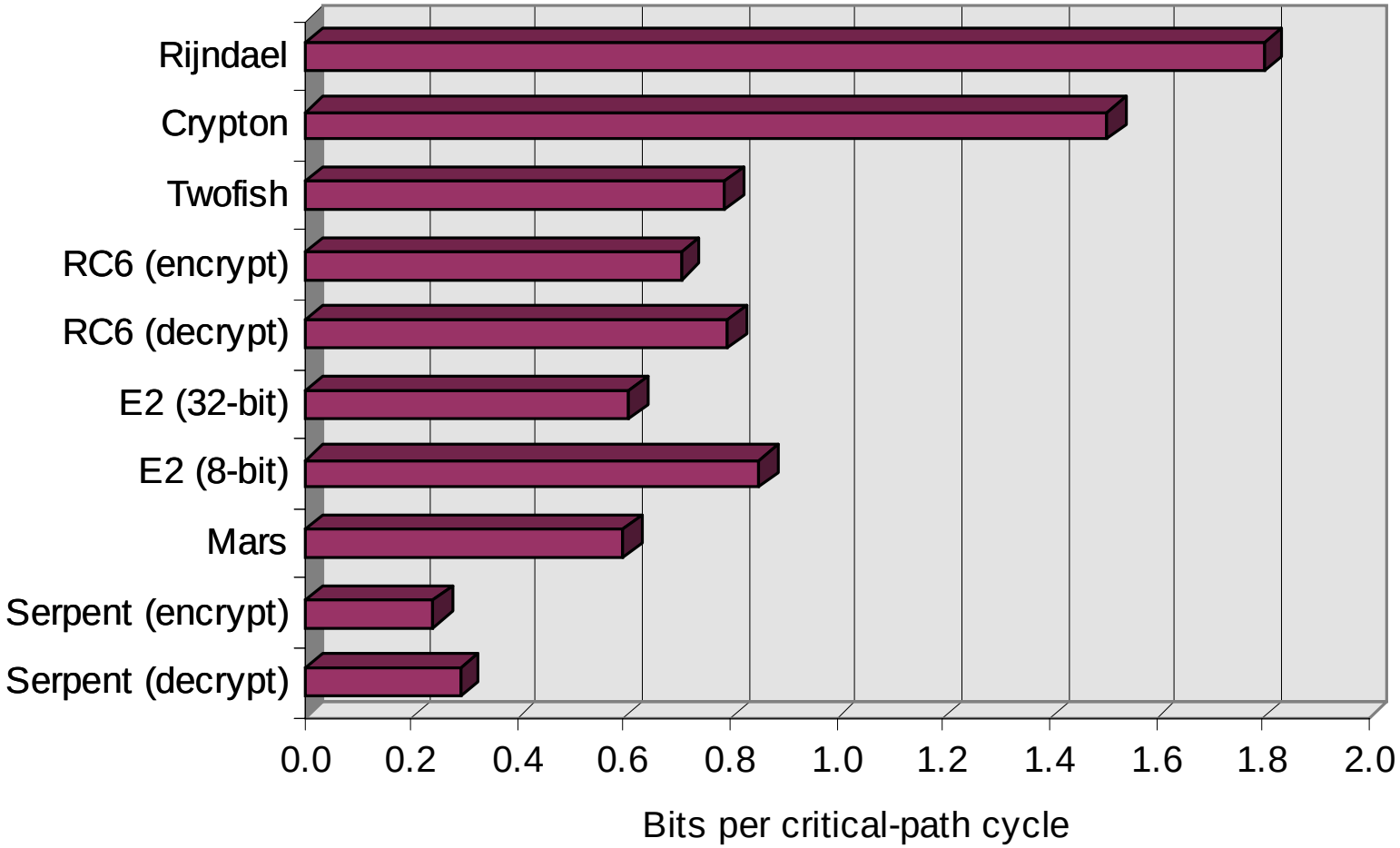
Critical-path example (2): RC6 decryption



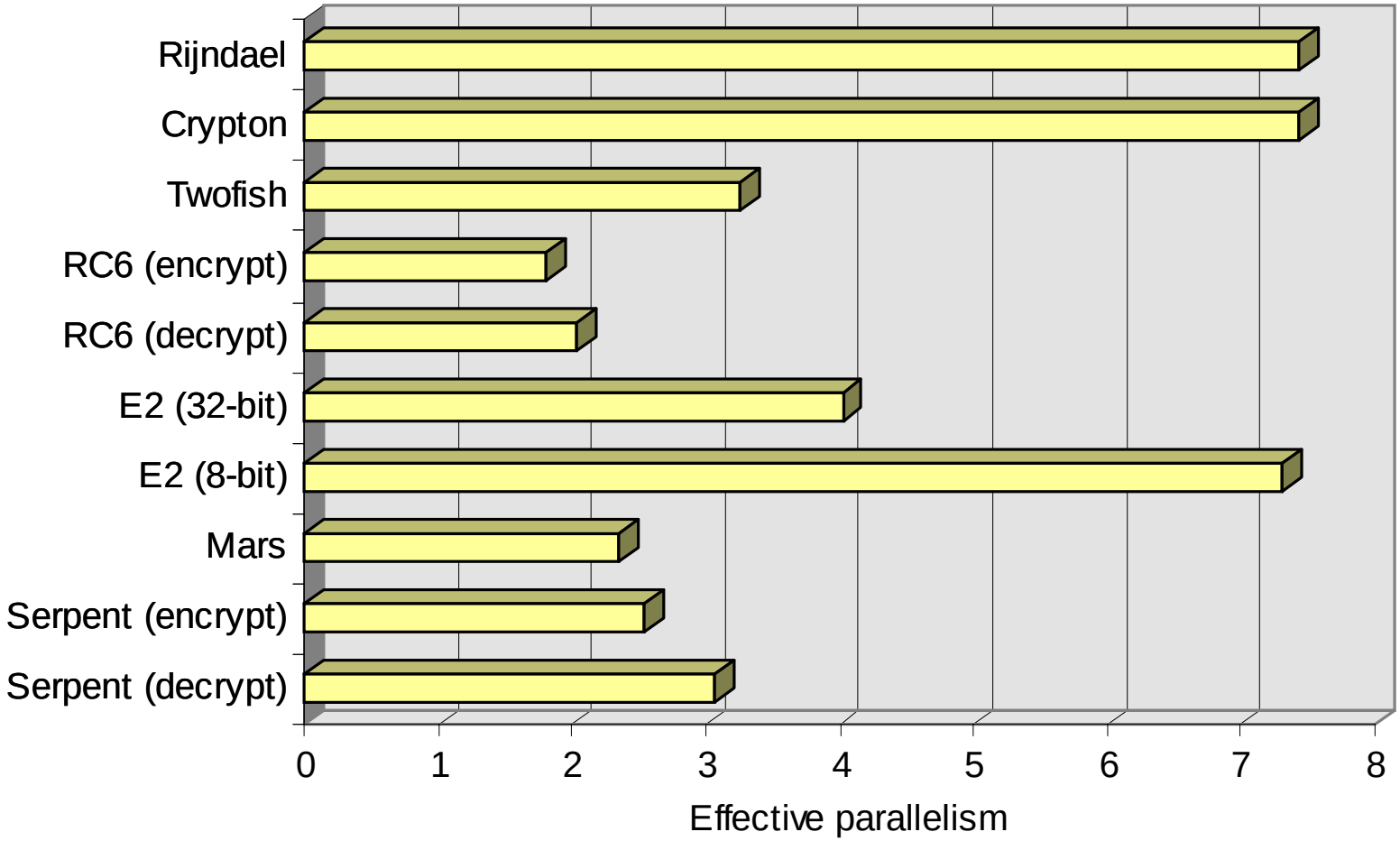
Critical path lengths



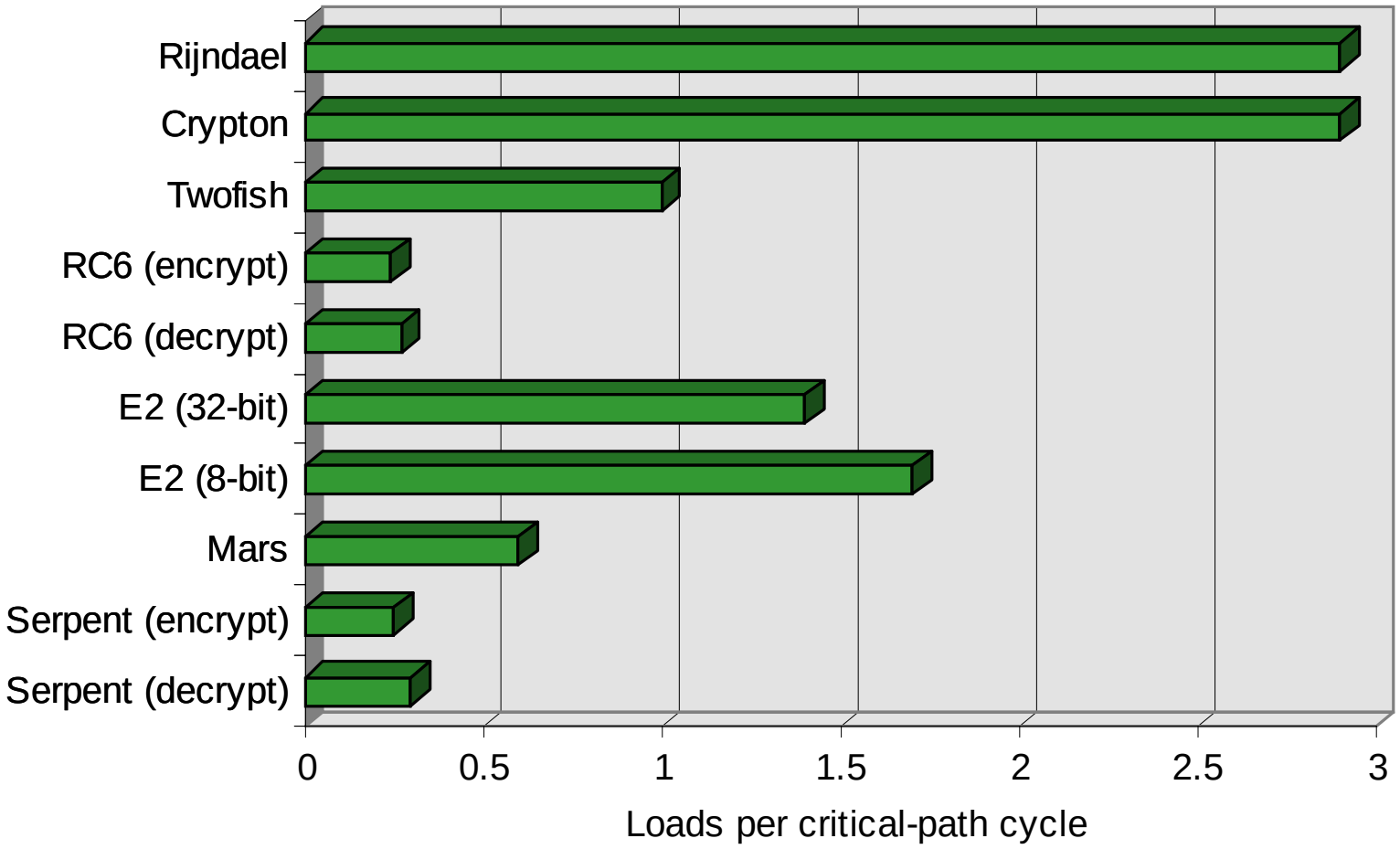
Throughput limits



Effective parallelism



Memory accesses

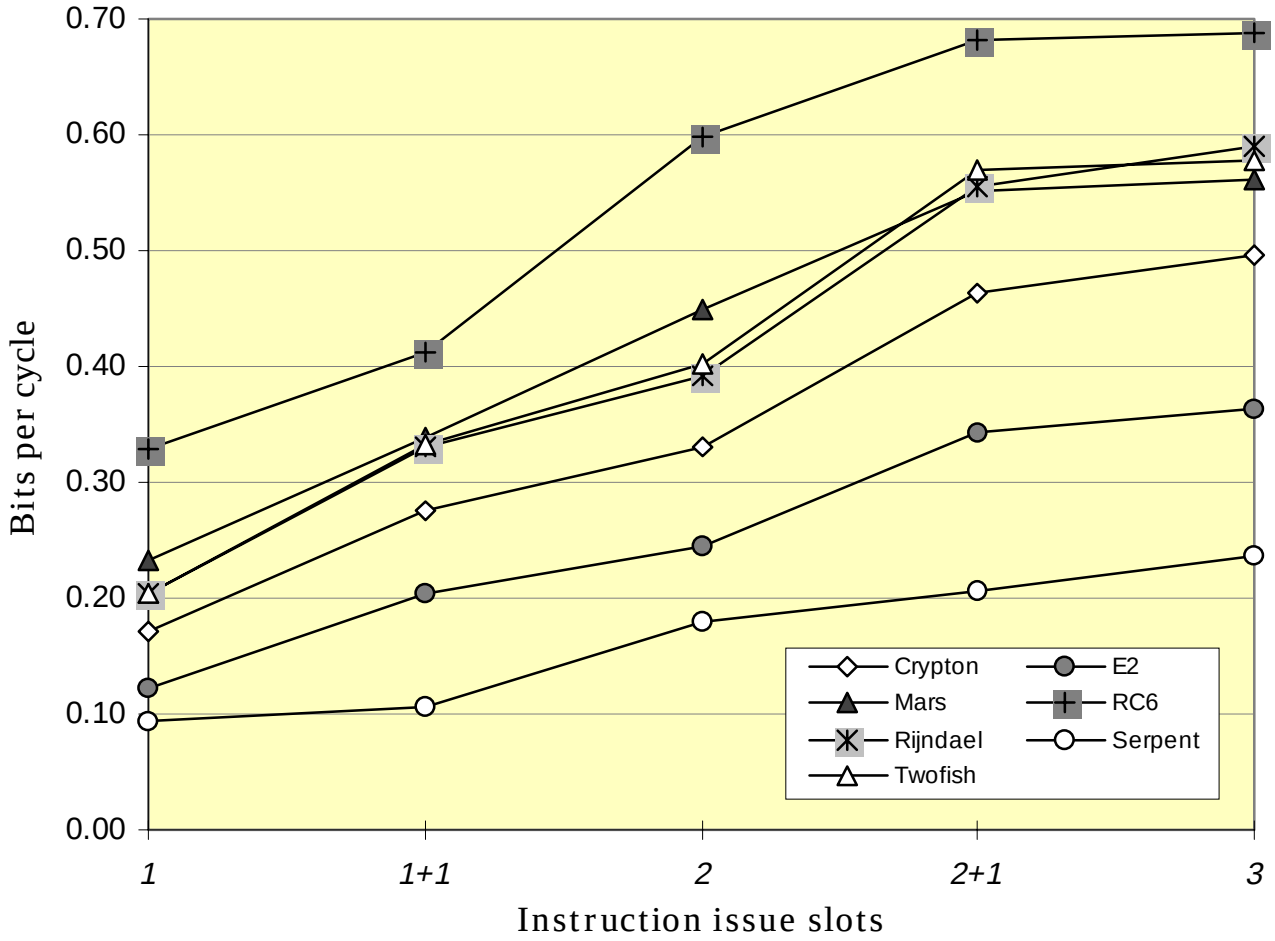


VLIW CPU Family Members

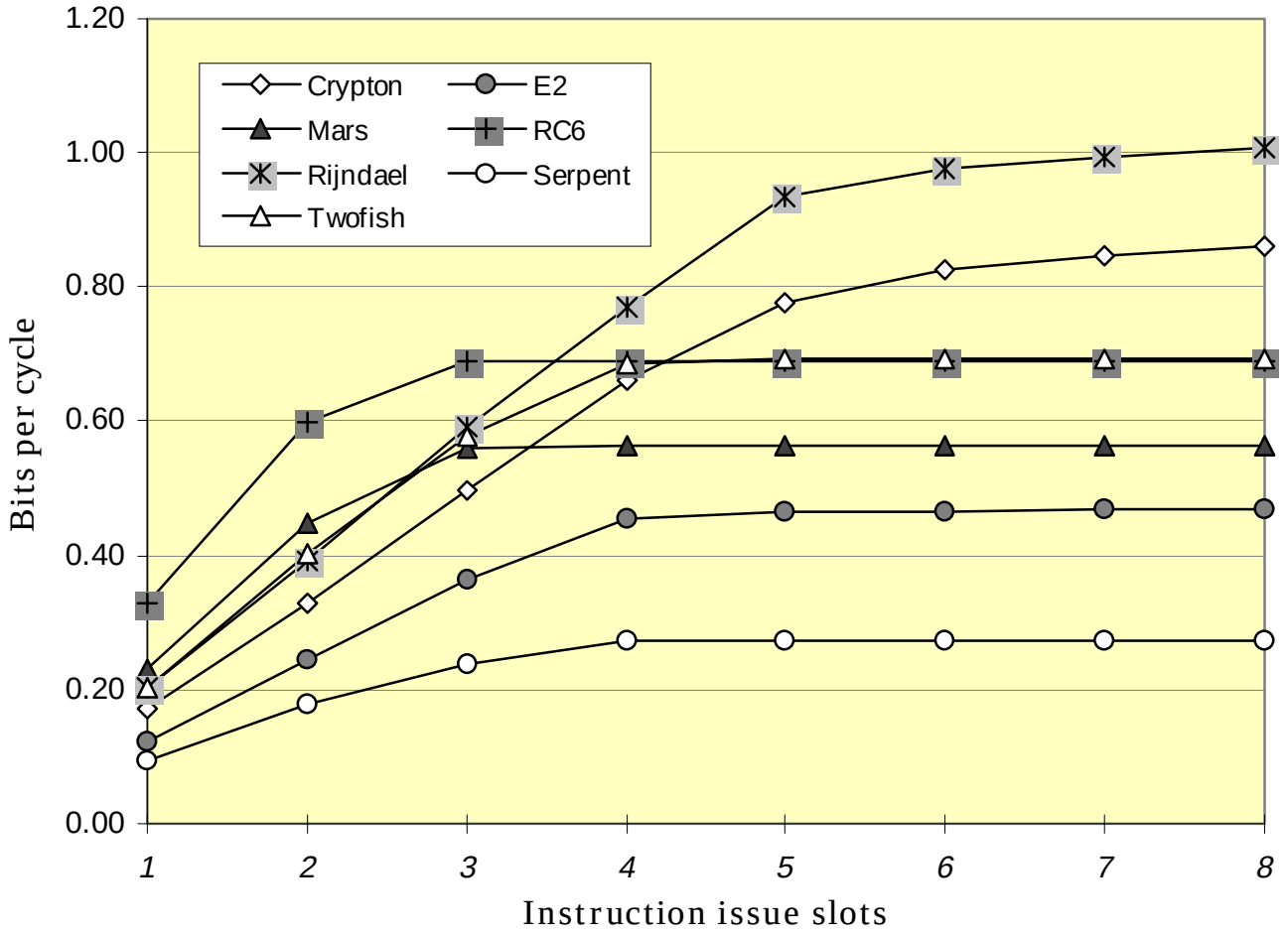
Instruction issue slots	*Disposition of functional units*							
	Slot 1	*Slot 2*	*Slot 3*	*Slot 4*	*Slot 5*	*Slot 6*	*Slot 7*	*Slot 8*
1	ALU, MEM							
1+1	MEM	ALU						
2	ALU, MEM	ALU, MEM						
2+1	ALU, MEM	MEM	ALU					
3	ALU, MEM	ALU, MEM	ALU					
4	ALU, MEM	ALU, MEM	ALU	ALU				
5	ALU, MEM	ALU, MEM	ALU	ALU	ALU			
6	ALU, MEM	ALU, MEM	ALU	ALU	ALU	ALU		
7	ALU, MEM	ALU, MEM	ALU	ALU	ALU	ALU	ALU	
8	ALU, MEM	ALU, MEM	ALU	ALU	ALU	ALU	ALU	ALU

Key: ALU = Arithmetic/Logic Unit, MEM = Load/Store Unit

Candidate performance versus execution resources (1)



Candidate performance versus execution resources (2)

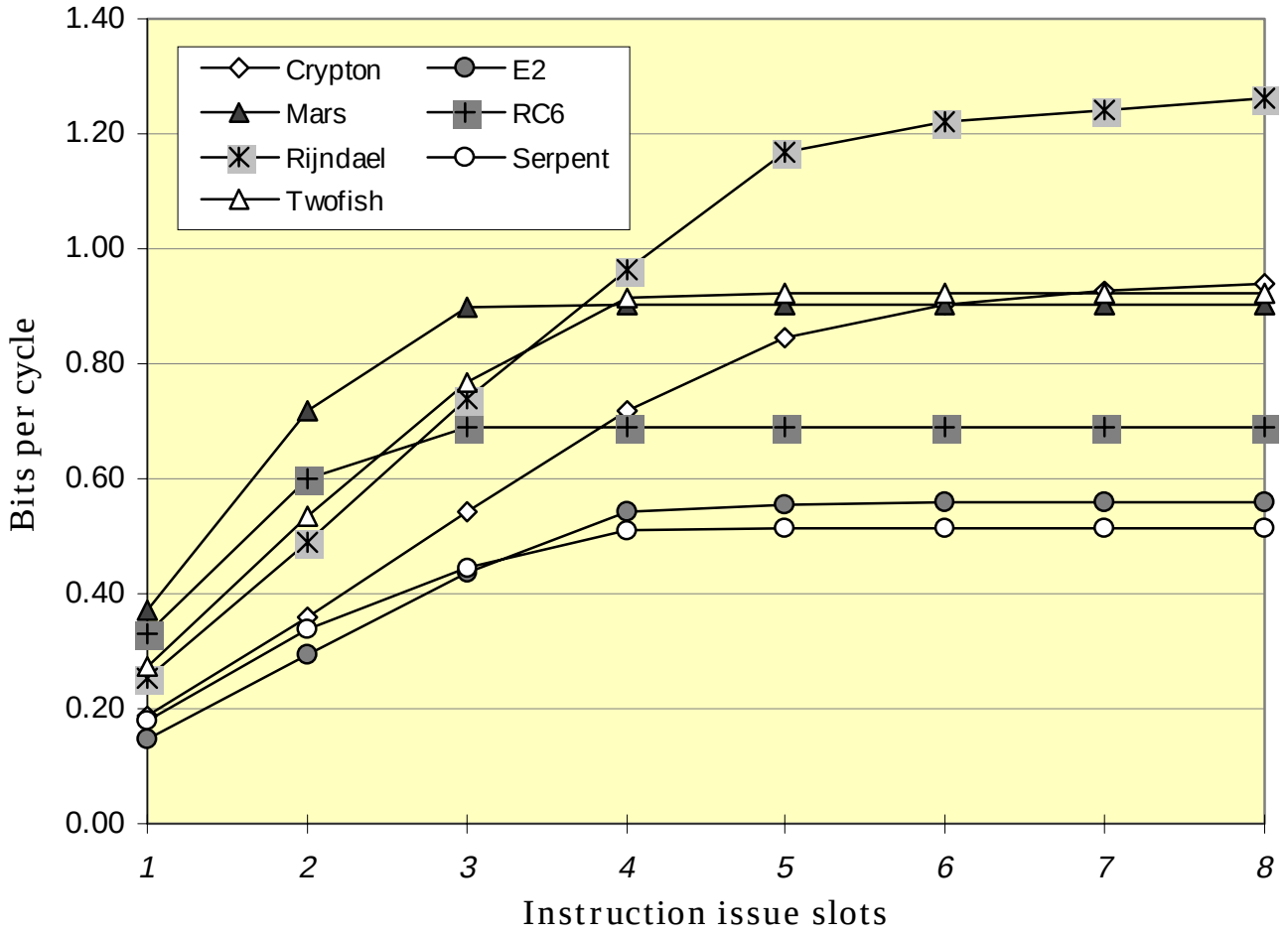


Normalization

Algorithm	*'Official' number of rounds (128-bit key)*	*Number of rounds for 'normalized' security†*
Crypton	12	11
E2	12	10
Mars	16+16	12+8
RC6	20	20
Rijndael	10	8
Serpent	32	17
Twofish	16	12

†. According to Biham

Normalized performance versus execution resources



Conclusions

- Our experiments have explored the likely performance of several algorithms on high-end processors of the future
- Our critical-path analysis correlates well with the practical case of code compiled for our hypothetical family CPUs
- At up to three execution units RC6 is the fastest algorithm
- Mars, Rijndael, and Twofish have virtually identical (second place) performance for up to three execution units
- At four or more execution units Rijndael takes the lead
- Crypton and Rijndael are the candidates most able to benefit from increasing instruction-level parallelism in CPUs